LISTing Newsletter

Newsletter of the Long Island Sinclair/Timex Users Group
(Incorporating N.Y.T.S.E.) **JUNE 1992 ISSUE**

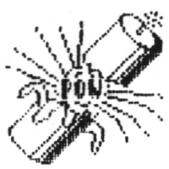
June 1992

SUN	MON	TUE	WED	THU	FRI	SAT
	1	2	3	4	5	6
7	8	9	10	11	12	13
[redacted]	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

DONT FORGET L.I.S.T. MEETING at 2pm, SUNDAY, June, 14

Table Of Contents

***** ** **********



SWAP & SHOP
EXPLOSIVE DEALS!

LISTing Policy *Annual Dues....$16.00*

One "sample" copy sent upon reciept of a large SASE.
Copies provided on EXCHANGE BASIS with other bona fide user groups. LISTing is published monthly except July and August by LIST (Long Island Sinclair Timex) Group, a non profit user group.

We are always looking for articles, programs, reviews, etc. to keep our members informed and entertained. You maintain full copyright.

LIST disclaimes any responsibility for any damage you may do to your computer as a result of reading any articles in LISTing.

LIST OFFICERS

PRES. HARVEY RAIT
TRES. ROBERT MALLOY
COR.SEC. JOHN PAZMINO
EDITOR. FRED STERN
LIBR. TOM SKAPINSKI

PLEASE SEND INQUIRIES TO:
LIST
MR. HARVEY RAIT
5 PERI LANE
VALLEY STREAM, N.Y. 11581

PLEASE SEND SUBMISSIONS TO:
LISTING
MR. FREDERIC STERN
214 ROBERTS ST.
HOLBROOK, N.Y. 11741

NYTSE

NYTSE MEETS ON MONDAY THE WEEK AFTER THE LIST MEETING AT;
MISS KIMS RESTAURANT
PARK AVENUE SOUTH
BETWEEN 21 ST. AND 22 ST.
MEETINGS START 7:30 PM.

COMING EVENTS:

JUN. 14, 1992 LIST MEETING AND SWAP MEET.
JUN. 23, 1992 NYTSE MEETING

MEETING MINUTES
REPORTED BY:
FRED AND MICHAEL STERN

MAY. 17, 1992

HARVEY CALLED THE MEETING TO ORDER AT 2:30PM.

NEW BUSINESS

WE RECEIVED A FEW LETTERS WHICH INCLUDED 1 NEW MEMBER. THIS BRINGS THE MEMBERSHIP TO 40.

WE WERE HAPPY TO HAVE ALVIN BRANDON IN ATTENDENCE AT THE MEETING. ALVIN EXPLAINED THAT HE HAS A NEW JOB AS A N.Y.C.T.A. BUS DRIVER. HE WORKS SATURDAYS AND SUNDAYS, WHICH PREVENTS HIM FROM COMING TO MEETINGS.
WE ALL WISH ALVIN MUCH SUCCESS IN HIS NEW ENDEVOR AND HOPE HE FINDS THE TIME TO COME TO FUTURE MEETINGS.

SWAP MEET

OUR NEXT MEETING WILL INCLUDE OUR BEFORE THE SUMMER SWAP MEET. BRING ALL YOUR TIMEX/SINCLAIR SURPLUS GOODIES TO SELL AND MONEY TO PURCHASE BARGINS. COME EARLY FOR THE BEST BUYES. IF PAST HISTORY IS AN INDICATION, THIS SWAP MEET SHOULD BE A BIG SUCCESS.

OTHER HAPPENINGS

A ROUND TABLE DISCUSSION DEVELOPED ABOUT HAPPENINGS IN THE QL WORLD. A QL EMULATOR FOR P.C. IS IN THE WORKS AND HOPEFULLY WILL SOON BE ON THE MARKET. WORD IS OUT THAT AMSTRAD HAS BEEN APPROACHED REGARDING THE PURCHASE OF THE RIGHTS FOR QL-DOS.

NOT BEING A QL USER, THIS IS AS CRYPTIC FOR ME TO WRITE AS FOR YOU TO READ, WHICH BRINGS ME TO MY POINT. LISTING NEEDS A QL REPORTER. SOMEONE WHO KNOWS THE MACHINE, HAS A FINGER ON THE QL NEWS PULSE, AND LOVES TO WRITE ABOUT IT AS I LOVE TO WRITE ABOUT THE TS1000.

THE QL IS A GREAT MACHINE, AND LIST HAS MANY QL INTERESTED USERS. IF YOU HAVE THE RIGHT STUFF TO BE THE LIST QL REPORTER PLEASE CONTACT ME: FRED STERN - LISTING EDITOR.

DEMONSTRATION

BOB MALLOY DEMONSTRATED A PROGRAM WHICH ANALYZES STOCK DATA. THE PROGRAM ORIGINALLY WAS DESIGNED FOR THE TS1000, BOB REDEVELOPED IT FOR THE QL. BOBS BIGGEST PROBLEM WAS CONVERSION OF THE COMMANDS TO SAVE: STRING DATA.
THE PROGRAM GIVES SHORT TERM INFORMATION FOR UP TO 60 STOCKS AT ONE TIME. THE PROGRAM CAN ALSO GENERATE A GRAPH OF STOCK VALUE VS TIME.

CLASSIFIEDS

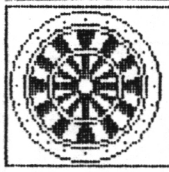
THIS CLASSIFIED SECTION IS AVAILABLE TO ALL LIST MEMBERS FREE OF CHARGE.
THE ONLY RESTRICTION IS THAT IT IS TO BE USED ONLY FOR THE SEEKING, SELLING OR SWAPPING OF SINCLAIR, TIMEX OR MICROACE COMPUTER EQUIPMENT, PERIPHERALS AND SOFTWARE.
LISTING, LIST, AND ITS OFFICERS DO NOT ENDORSE, WARRANTY, OR GUARANTEE ANY OF THE ITEMS LISTED IN THIS CLASSIFIED SECTION

I HAVE MERCHANDICE OF INTEREST TO THE TIMEX/SINCLAIR HACKER AND EXPERIMENTER. SEND A S.A.S.E TO
VAN S. VANGOR
346 C. RETREAT ROAD
ISLAND FALLS, MAINE. 04747
FOR MY LIST AND PRICES.

THE FOLLOWING PUBLICATIONS ARE AVAILABLE ONLY THROUGH LIST:

ZX-81/TS1000 TECHNICAL TIDBITS
TECHNICAL TIDBITS PART II
SAVINGS AND LOAD OF THE TIMEX COMPUTER
$4.00 EACH.

Continue Page 4



ON TARGET
HINTS&TIPS

SAVINGS AND LOAD

K5XY

I have just been sent a very interesting booklet from L.I.S.T., the Long Island Sound Sinclair-Timex Group. It is entitled *Savings and Load of the Timex Computer.* I found it very interesting.

It is another in their series of booklets designed to collect in one place some of the useful technical information about the ZX81 series of computers. This booklet concentrates on the problems of saving and loading programs with tapes. One of the reasons our computers are so inexpensive is that Sinclair decided to use audio tape as the storage medium. In that way inexpensive media was readily available and, because ordinary audio recorders were cheap and available, the computer did not have to contain the required electronic and mechanical parts to record programs. The disadvantages, as we all know, were that the method is slow and unreliable. This booklet collects information which helps decrease these problems.

The lead off reprint is the best in the publication. It deals with tape recorder head alignment and provides as complete coverage of that subject as you can expect from four information packed pages. It even tells a bit about the actual signals put on the tape by the computer. Included are several short articles dealing with the amplitude of the recorded signals. There are suggestions for increasing the amplitude and suggestions on how to prevent too much amplitude.

Interestingly enough "turn down the volume control" is not one of these latter suggestions. This last sentence really is not fair. One article did contain the sentence "For best results use a lower volume setting when using the SIGNAL BOOSTER". Actually in some circumstances it does make sense to use a lower volume setting with a signal booster. If a low power tape recorder is run wide open its frequency response and distortion product output will suffer thus changing the shape of the data pulses. Some boosters can be thought of as impedance matching devices too.

The final topics covered are some short hints for better operation and a program to speed up the data transfer rate. Two short 1 K games are included too.

All of the material in the booklet consists of reprints from hard to get publications. One contribution even comes from long time QZX reader WA4BQE. The booklet consists of 13 single sided xeroxed sheets and is obtainable from

L.I.S.T.

5 Peri Lane

Valley Stream, NY 11581.

CLACKAMAS COMPUTER APPLIED TRAINING SOCIETY

THE PLOTTER

COMPUTING THE FUTURE--
IN CLACKAMAS COUNTY

BITS & BYTES

by: Rod Gowen

NEWSLETTER EXCHANGE-

I have had a LOT of reading to do in the last few weeks. It seems as though the exchange newsletters pile up rather quickly if I don't get to them right away! I have scanned and read over 30 newsletters in 2 weeks. We have lost a couple and gained some new ones over the past year or so. I really cannot see any newsletter folding for "lack of news" as some may say. I see a real wealth of information that NEEDS to be passed on to our own group members and the readers of THE PLOTTER who may not have access to these other fine newsletters.

A couple of the best newsletters that come in on a regular basis, be it monthly or bi-monthly, are: SINC-LINK from the TORONTO USERS CLUB and L.I.S.T. from the LONG ISLAND, NY group. I recommend that anyone who wants to see the TS community continue subscribe to as many of the existing newsletters as they can possibly afford!

»»»

EDITORIAL
****

THANKS ROD FOR YOUR GRACIOUS COMPLIMENT. WE TRY OUR BEST TO KEEP UP WITH THE PLOTTER. IT IS NOT EASY.

ANOTHER EXCELLENT, AND THE NEWEST PUBLICATION IN THE TIMEX COMMUNITY IS ANDRE BAUNES ZX-81. (SEE BELOW)

ZX-81

10 YEARS LATER

ALL

THE NEW NEWSLETTER FOR THE SINCLAIR ZX81 / TIMEX 1000.

ALL YOU SEE, AND WILL SEE, HAS BEEN CREATED WITH, AND ONLY, A ZX81, A 16K RAMPACK AND A TIMEX 2040 PRINTER.

IF YOU WISH TO RECEIVE
A PRINT-OUT, A CASSETTE,
AN INFORMATION, A REPLY
OR THE NEXT MONTH ISSUE
OF THIS NEWSLETTER SEND
A SELF-ADDRESSED ENVELOPE
TO: ANDRE BAUNE
304 SCOTT,
CHATEAUGUAY, QUEBEC
CANADA J6J 4H5

From Page 2

A FINAL WORD
****
MY NAME IS FRED STERN AND I AM THE EDITOR OF THIS EDITION OF LISTING.

AS I STATED ABOVE, LISTING NEEDS A QL REPORTER. I ALSO NEED NEW ORIGINAL ARTICLES FOR PUBLICATION. THIS IS YOUR NEWSLETTER, PLEASE SUPPORT AND CONTRIBUTE TO IT.

SPECIAL THANKS ARE EXTENDED TO; MICHAEL STERN, AND TOM SKAPINSKI WHO DID A *DYNOMITE*JOB IN HELPING TO GET THIS NEWSLETTER OUT.

SEE YOU ALL AT THE NEXT MEETING.

Journey to the centre of the ROM

Andrew Hewson delves into the heart of the Z80 to unearth some useful routines.

SINCLAIR Research have manufactured three machines all based on the Z80 microprocessor: the ZX-80, the ZX-81 and the ZX Spectrum, and it is interesting to observe the development from one machine to the next.

A consequence of that understandable policy of developing one ROM from its predecessors is that some features which were necessary or desirable in the earlier version may be retained in the later version because the relevant code is known to work even though they impose constraints on the new design. This is, indirectly, the answer to the following question from John Blackwood of Wakefield who asks **"Why is 9999 the largest line number permitted on the ZX-81?"**

At first sight the limitation to 9999 seems quite illogical because 9999 is not a 'round number' as far as the Z80 microprocessor is concerned. A more logical upper limit might be 255 because that is the maximum integer which the machine can store in a single memory location — or byte. However, many users could be expected to write programs containing more than 255 lines and so a greater limit is desirable. The next highest 'round number' is 65535 which is the largest integer which the computer can store in two consecutive bytes. So why limit line numbers to 9999 when 65535 could be used just as easily?

The reason appears to be that by limiting line numbers in that way and by manipulating the numeric codes for variables the ZX-81 has a device for distinguishing lines in the program area from variables in the variables area.

To understand the mechanism at work, consider the binary representation of 9999. Line numbers are held with their most significant byte first. That is contrary to the usual Z80 convention so we can assume that the manufacturers had some special motive in choosing that arrangement. Hence line number 9999 is held as a byte containing 39 followed by a byte containing 15 because

39 * 256 + 15 = 9999

The bit pattern of the byte, obtained by converting 39 to binary, is 00100111. Notice that the three most significant bits — bit numbers 7, 6 and 5 are set to 0, 0 and 1 for this, the largest permitted line number. Hence bit numbers 7, 6 and 5 of the first byte of all permitted line numbers will be set to 0, 0 and 1, or in the case of line numbers less than 8192, they will be set to 0, 0, 0.

Now look at pages 172 to 174 of the *ZX-81 Basic Programming* manual and you will see illustrations of the different types of variables as they are represented in the variables area. In each case the first byte contains a numeric code related to the code of the letter which identifies the variable — or the code of the first letter of the variable name in the case of a number whose name is longer than one letter. The largest possible letter code is 63, the code for Z, which is still 00111111 in binary, and the smallest is 38, the code for A, which is 00100110 in binary.

Hence bits 7 and 6 are not needed when distinguishing between letter codes and as bit 5 is always set to one, the ZX-81 can use these bits to distinguish between the different types of variable. Three bits can be set in

2 * 2 * 2 = 8

different ways. Table 1 lists the eight ways and their interpretation.

It is strange that Sinclair should take such elaborate precautions to distinguish a line number from a variable because the same purpose could be served by comparing the address of the byte in question to the D-FILE or VARS system variables. It allows the ZX-81 to use the same routine, at 2546 to 2576, to step through memory to the 'next' line or the 'next' variable but that seems a small advantage.

It is certainly one of the

Table 1. This shows the meaning of the top three bits in the first byte of a program line or Basic variable in the ZX-81 and the Spectrum.

Bit pattern	Interpretation
000	Line number less than 8192
001	Line number between 8192 and 9999
010	String
011	Number with single character name
100	Array of numbers
101	Number with multiple character name
110	Character array
111	Control variable for a FOR-NEXT loop

Table 2. A Spectrum program to PRINT the characters with codes in the range 32 to 255 inculsive. Note that when the a register contains 255, the effect of the inc a instruction is the same as subtracting 255, ie a subsequently contains zero.

Decimal	Assembly	Code	Comment
62 32		ld a, 32	Load the a register with 32
245	Again	push af	Save a on the stack
215		rst 16	PRINT the character
241		pop af	Retrieve a from the stack
60		inc a	Increment the a register
32 250		jr nz , Again	Jump to PRINT next character
201		ret	Return when a reaches zero

Table 3. A simple decimal loader for POKEing decimal numbers into the Spectrum printer buffer. To halt the program enter STOP (Symbol Shift A).

```
10   FOR I = 23296 TO 23551
20   INPUT J
30   POKE I , J
40   PRINT I , J
50   NEXT I
```

features which has been carried forward from the ZX-80 to the ZX-81 and then to the Spectrum.

We shall return to discussing Basic variables later but first a small but relevant digression is prompted by the following question from Patrick Higham of Manchester. He asks: **"Is there a simple method of printing characters on the Spectrum screen from a machine code routine?"**

Printing from machine code is very straightforward because the manufacturers have thoughtfully provided a routine in ROM to do all the hard work. The routine is called at address 16 decimal — 10 in hexadecimal — and should be accessed using the special Z80 machine code instruction

RST 16

That instruction, for some reason which has never been adequately explained, is called a 'restart' — hence the RST abbreviation — and is one of eight such special instructions. As far as the user is concerned it has the same effect as a CALL instruction except that only one byte instead of three is required to hold it.

The routine is entered with the A register set to the code of the character to be PRINTed and the appropriate character appears on the screen at the current PRINT position. All registers are preserved by the routine except the AF register pair and so in some circumstances it may be necessary to PUSH and POP AF before and after the RST instruction respectively.

The routine listed in table 2 demonstrates the use of RST 16 by using it to PRINT all characters with codes lying between 32 and 255 inclusive. Note that includes all the tokens so the routine demonstrates that command words like POKE, READ and DRAW can be PRINTed using RST 16 if required. The decimal codes for the routine can be loaded into the printer buffer using the decimal loader listed in table 3.

The RST 16 facility can also be used to control the screen format and layout character codes but a little care must be taken not to follow the INK, PAPER and other control codes by invalid numbers because otherwise error code K results. Some of those layout characters are extremely useful, for example

LD A , 13
RST 16

will PRINT an 'ENTER' character so that the current PRINT position will move to the start of the next line.

Of course, the PRINT routine at address 16 was not provided by the manufacturers solely for the benefit of users of the finished machine. The Spectrum ROM itself makes extensive use of the facility and so it is littered with RST 16 instructions. That goes some way to explaining the power of RST instructions. Every time one is used two bytes of memory are saved — the difference between the length of a CALL and a RST instruction — and more importantly the Z80 does not waste time calculating the address which is being called because it is implicit in the instruction. Hence RST is very useful for calling routines which are used frequently.

The call to RST 16 is an important part of the routine which is listed in table 4 in response to the following letter from Alan Procter of Windsor: **"Have you a routine to identify the variables existing in memory, identifying them as numerics, string simple or array?"**

The routine is rather longer than the ones I usually include in this column and so I recommend that an assembler program is used to load it into memory. Please note that the routine is not relocatable ie if the decimal codes are used it can only be loaded into the printer buffer starting at 23296.

The routine contains six subroutines which I have called setb, prtvar, prtdol, addlen, lonvar, addsix and prtype. Those perform the following functions:

setb There are six different types of Basic variables and this subroutine looks at the first byte of the current variable and puts the corresponding value into the b register. Table 5 shows the types and the value of b.

prtvar Each type of Basic variable except one has a single character name. This subroutine decodes the character code by successively subtracting 32 and PRINTs the result.

prtdol If the current variable is a string or a string array, this subroutine PRINTs a dollar sign to follow the single character name.

addlen The two bytes following the single byte name of strings, string arrays and character arrays contain a number equal to the number of byte used to store all the data in the variable. This routine adds this number to the pointer in hl so that the next variable can be found.

lonvar This routine PRINTs up to ten characters of the name of a numeric variable with a multiple character name and moves the hl pointer to the end of the variable name.

addsix Six bytes are used to hold the data in numeric variables. This subroutine adds six to the hl pointer so that the next variable can be found.

prtype This subroutine TABs to column 16 using the ROM routine at OA5Fh and then PRINTs the appropriate variable type as determined from the value in the b register from the list of types held in the 'table' at the end of the routine.

Table 5. The codes placed in the b register by the "setb" routine in table 4 and the corresponding variable types.

b register	Type of variable
1	String of characters
2	Simple numeric variable
3	Numeric array
4	Numeric variable with multiple character name
5	String array
6	Loop counter

Table 4 continued

Decimal	Assembly Code	Comment
99	defb 99	
128	defb 128	
83	defb 83	
116	defb 116	
114	defb 114	
105	defb 105	
110	defb 110	
103	defb 103	
32	defb 32	
97	defb 97	
114	defb 114	
114	defb 114	
97	defb 97	
121	defb 121	
128	defb 128	
76	defb 76	
111	defb 111	
111	defb 111	
112	defb 112	
32	defb 32	
99	defb 99	
111	defb 111	
117	defb 117	
110	defb 110	
116	defb 116	
101	defb 101	
114	defb 114	
128	defb 128	

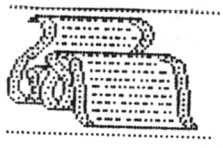
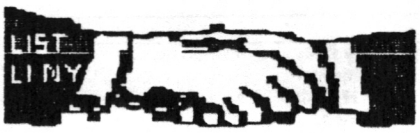


Table 4. A Spectrum routine to PRINT the names and types of all the current Basic variables.

Decimal	Assembly Code		Comment
42 75 92		ld hl, (23627)	VARS to hl
126	tend:	ld a, (hl)	128 is the end of VARS marker
254 128		cp 128	
200		ret z	
205 111 91		call setb	Set b register to 1 to 6
205 55 91		call prtvar	Print the name of the variable
205 67 91		call prtdol	Print the dollar sign if necessary
203 64		bit 0,b	Is b even?
40 5		jr z,even	If so jump
205 125 91		call addlen	No, addlen skips over the data
24 22		jr prtres	Skip to prtres
203 80	even:	bit 2,b	b is even. Is it 2?
40 15		jr z, two	If so jump
203 72		bit 1,b	Is b 4?
40 8		jr z, four	If so jump
213		push de	b is 6 so this is a loop counter
17 13 0		ld de, 13	Skip over 13 bytes
25		add hl,de	
209		pop de	
24 3		jr two	
205 141 91	four:	call lonvar	This is a long variable
205 134 91	two:	call addsix	Skip over remaining six bytes
205 77 91	prtres:	call prtype	Print type of variable
62 13		ld a,13	Output the enter character
215		rst 16	
24 204		jr tend	
197	prtvar:	push bc	This routine decodes and prints the
198 32		add 32	(first letter of the) variable held
16 3	tryb:	djnz decb	in the a register
215		rest 16	
193		pop bc	
201		ret	
214 32	decb:	sub 32	
24 247		jr tryb	
203 64	prtdol:	bit 0,b	This routine prints a dollar sign
200		ret z	if b contains 1 or 5
203 72		bit 1,b	
192		ret nz	
62 36		ld a,36	
215		rst 16	
201	ret		
229	prtype:	push hl	This routine prints the appropriate
197		push bc	variable type taken from the table
33 168 91		ld hl,table	controlled by the value in b
16 7	dec:	djnz nexlet	
229		push hl	
205 95 10		call 0a5fh	Tab to column 16
225		pop hl	
24 8		jr out	
126	nexlet:	ld a, (hl)	Decrement b if the next letter has
35		inc hl	code 128
254 128		cp 128	
40 241		jr z,dec	When b is zero the correct entry
24 248		jr nexlet	has been found
126	out:	ld a, (hl)	Print each letter in turn, first
254 128		cp 128	checking for the 128 terminator
40 4		jr z,endpr	
215		rst 16	
35		inc hl	
24 247		jr out	
193	endpr:	pop bc	Restore bc
225		pop hl	and hl
201		ret	before returning
6 1	setb:	ld, b,1	This routine sets the value of b in
14 91		ld c,91	range 1 to 6 according to the value
185	nexb:	cp c	in the first byte of the variable name
216		ret c	
4		inc b	
121		ld a,c	
198 32		add 32	
79		ld c,a	
126		ld a,(hl)	
24 246		jr nextb	
213	addlen:	push de	This routine is used to skip over the
35		inc hl	body of the data for numeric arrays,
94		ld e,(hl)	strings and string arrays
35		inc hl	
86		ld d,(hl)	
35		inc hl	
25		add hl ,de	
209		pop de	
201		ret	
213	addsix:	push de	Used for simple variables and long
17 6 0		ld de,6	variables to skip over the body of
25		add he ,de	the data
209		pop de	
201		ret	
197	lonvar:	push bc	This routine prints up to ten
6 10		ld b,10	characters from a long variable name
35	nexlon:	inc hl	
55		scf	
126		ld a,(hl)	
254 128		cp 128	
48 12		jr nc,endlon	
215		rst 16	
16 246		djnz nexlon	
35	aglon:	inc hl	Discard any remaining characters
55		scf	after the tenth character has been
126		ld a,(hl)	printed
254 128		cp 128	
48 5		jr nc,endl	
24 247		jr aglon	
214 128	endlon:	sub 128	
215		rst 16	
193	endl:	pop bc	
201		ret	
83	table:	defb 83	
116		defb 116	
114		defb 114	
105		defb 105	
110		defb 110	
103		defb 103	
128		defb 128	
83		defb 83	
105		defb 105	
109		defb 109	
112		defb 112	
108		defb 108	
101		defb 101	
32		defb 32	
118		defb 118	
97		defb 97	
114		defb 114	
105		defb 105	
97		defb 97	
98		defb 98	
108		defb 108	
101		defb 101	
128		defb 128	
78		defb 78	
117		defb 117	
109		defb 109	
101		defb 101	
114		defb 114	
105		defb 105	
99		defb 99	
32		defb 32	
97		defb 97	
114		defb 114	
114		defb 114	
97		defb 97	
121		defb 121	
128		defb 128	
76		defb 76	
111		defb 111	
110		defb 110	
103		defb 103	
32		defb 32	
110		defb 110	
117		defb 117	
109		defb 109	
101		defb 101	
114		defb 114	
105		defb 105	



Spectrum To IBM Emulation Program

Sinclair Spectrum 48K Z80 Emulator for the IBM PC, or Clones, 80386SX, VGA

Well I got a copy from a friend, he told me it worked. What this meant I actually didn't know. But my curious mind wanted me to try the thing out and see what it did and how well it did or did not work. I was hoping it would actually work. Well it does! What I found was on the three PC's I had access to, an XT 10MZ,8088; AT 16MZ, 80286; and a 33MZ, 80386 it worked with the following results.

As you might expect the emulator and programs that were loaded ran very slowly on the XT machine but it did work. (However I was not able to realiably load from tape to this machine.) And this machine only has low rez. EGA 320*200. So VGA video is not necessary, however it may be used if that is what you have.

On the AT 16 MZ machine the emulator and loaded programs ran fairly well speed wise but slower than a real Spectrum. The 386 machine appears to be a little faster than the real Spectrum it emulates, and can load Spectrum tapes. Not all programs will load but many will. I believe the ones with hyper loaders give trouble.

I found putting the emulator on the hard drive was the only way to load files into it. So first hook up the cable and tape recorder. Then load the emulator disk. Run it, load programs from tape, (*) save them using to the hard drive using F2 This saves the program running as an NMI file and gives it a name; SNAP001.SP, which can be renamed from DOS. The program automatically increments the number each time you use F2.. Loading a program back is accomplished by entering SPECTRUM SNAP001.SP at the C:> prompt. The instructions for this program are on the emulator disk. You can print them out for further information. Since the original instructions were in Spanish a translation was done. The work of translating was done by Ron Hopkins-Lutz, the scanning was done by Greg DuPuy, the emulator was written by Pedro Gimeno. This review is by Tom Skapinski. The Greater Cleveland Sinclair Users Group also must be given credit. Pardon

me if I don't mention evryone that I should. I really am working quite independently here and offer this review only to kindle interest and to let members know of the existence of this software.

If you are interested in seeing your old favorite Spectrum programs again. Only this time on your IBM or clone at home or at work. You may be able to download this program from PC-OHIO, GEine, CompServe ,Clevevland FreeNet or by writing to me.

If you choose to ask me for a copy you will get a sample program that was ported over to the PC from the Spectrum tapes as room on the disk allows. (Also include some money for duplication and mailing costs) (Suggestion: $3.00)
The programs that I have on the hard drive are, Critical, Fall Guy, Flag, Fred, Kong, Maziacs, Miner, Mugsy, Pitfall II, PSSST, Pyramid, Pystron, Saboteur, Skiing, Spectres, Splat, Tirnanog. You may have noticed some of the above titles are not the correct names but shortened ones that give you enough information to know which ones they are if you are familiar with the original titles.

I thought of offering these program to people but decided that there are such things as copyright laws. and I don't know what I can or can't do in regard to these old Spectrum programs. However I also have some of the converted graphics from Andre Baune's newsletter. These are the ones I addapted to the TS-2068, and now they can be seen on the IBM PC or clone. Incidently these can be printed to an Epson Compatable printer by the emulator. It supports the COPY comand like the TS-2040.

Write to: Tom Skapinski
7 Atkinson Lane
Coram NY 11727-3004 U.S.A.

If anyone else has been busy with this project please send me a letter telling of your experiences good or bad and then we can pass along info that comes in and hopefully we can help anyone else that is working with this emulator.

(*) Yes I did say LOAD from tape!
First make a simple cable; EARphone Jack to a 25 pin Male parallel connector. The instructions are on the in a file with the program. And it is easy to make, just one single conductor shielded cable. Center conductor to pin 13, and shield to pin 25. Start with a lov volume setting and work your way up untill loading becomes reliable.

I hope you found this article to be of interest. And encourages an extended life to your favorite Spectrum programs.

by Tom Skapinski L.I.S.T. Users Group
5/24/92

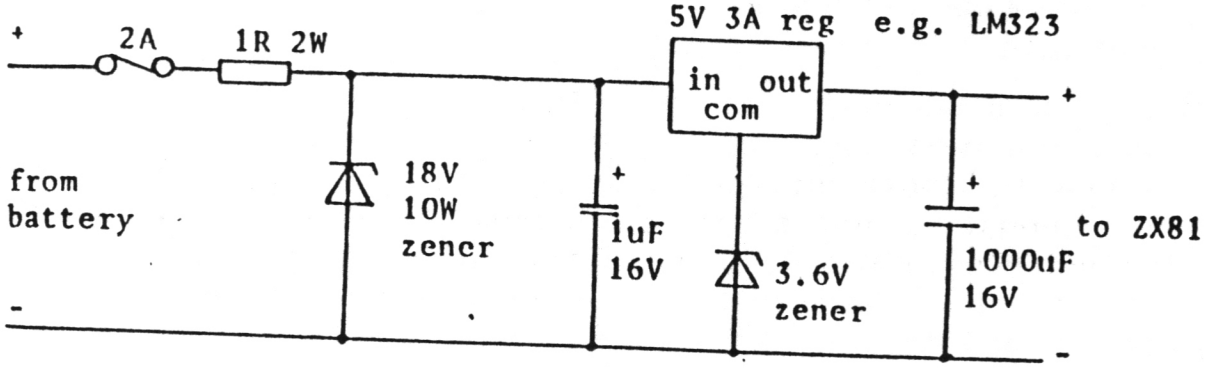
Hardware Hints

This section suggests some ways to improve the ZX81 hardware. Memory and I/O circuits are given at the end of this chapter.

Battery Power

Since portable televisions are available which will run off a 12 volt car battery, it would be nice to be able to run a ZX81 from the battery as well, for mobile computing perhaps, or for work during a power cut.

But, a car battery gives nominally 12 volts, which can rise to almost 15 volts when it is being charged heavily, and this is too high for the ZX81. To reduce the voltage to an acceptable level, use the circuit shown below, which will supply sufficient current (at least 1.2A) to drive a ZX81 fitted with a 16K RAM pack and the ZX printer.



The LM323 should be mounted on a heatsink; a piece of 1.5mm (1/16") aluminium 5cm x 8cm (2" x 3") is sufficient. The fuse, 1 ohm resistor and the 18 volt zener protect against voltage spikes coming from the battery lines, and also against reverse polarity connection, the 18V zener will not normally get hot so it does not need a heatsink.

This circuit is also useful if you suffer from frequent mains supply interruptions, which will 'zap' your program easily, as running your system from a car battery - which can either be recharged periodically or trickle charged all the time - should overcome the problem.

Cool it

All electronic equipment works more reliably if it is kept cool. Drilling a row of 1/4" holes along the rear face of the top moulding and some more in the base of the ZX81, near the front, will make a noticeable difference to the temperature inside the case. But take the case apart first - there are 5 fixing screws in the base, three being hidden by the feet - and carefully remove the printed circuit board first to avoid damaging it while drilling.